CHAPTER 2

Configuring X.25

This chapter describes how to configure connections through X.25 networks, including Link Access Procedure, Balanced (LAPB) connections. LAPB procedures are presented first, for those users who only want to configure a simple, reliable serial encapsulation method. For a complete description of the commands discussed in this chapter, refer to the *Protocol Translator Command Reference* publication. For more information about X.25 and LAPB, see the *Internetworking Technology Overview* publication.

Cisco's Implementation of LAPB and X.25

A group of specifications published by the CCITT (French acronym translating to International Telegraph and Telephone Consultative Committee) defines *recommendations*, or specifications, for X.25. The CCITT specifications basically specify connections between data terminal equipment (DTE) and data circuit-terminating equipment (DCE) for remote terminal access and computer communications. The X.25 specifications include LAPB as the data link layer protocol and X.25 as the network layer protocol, or packet layer protocol (PLP) as it is also known. The CCITT updates its specifications every four years, and the specifications dated 1980 and 1984 are the most common versions currently in use. Additionally, the International Organization for Standardization (ISO) has published ISO 7776:1986 as an equivalent to the LAPB standard, and ISO 8208:1989 as an equivalent to the CCITT 1984 X.25 Recommendation. Our X.25 software follows the ISO standards and the CCITT 1984 X.25 Recommendation, except for its Defense Data Network (DDN) and Blacker Front-End Encryption (BFE) encapsulation software, which is based on the CCITT 1980 X.25 Recommendation.

Briefly, Cisco Systems' X.25 software provides the following capabilities:

- X.25 Level 2, or LAPB (Link Access Procedure, Balanced)—LAPB is a data encapsulation protocol that operates at Level 2 (the data link layer) of the OSI reference model. LAPB specifies methods for exchanging data (in units called *frames*), detecting out-of-sequence or missing frames, retransmitting frames, and acknowledging frames.
- LAPB datagram transport—Protocol datagrams (IP, DECnet, AppleTalk, and so forth) are carried over a reliable LABP connection or datagrams of these protocols are encapsulated in a proprietary protocol and carried over a LAPB connection. The LAPB connection can carry a single protocol or multiple protocols, depending upon the configuration.
- X.25 datagram transport—Protocol datagrams (IP, DECnet, AppleTalk, and so forth) are encapsulated inside packets on an X.25 virtual circuit. Mappings between X.25 addresses and protocol addresses allow these datagrams to be routed through an X.25 network, thereby allowing an X.25 public data network (PDN) to transport local-area network protocols.

- X.25 switch—X.25 calls can be routed based on their X.25 addresses either between serial interfaces on the same protocol translator (local switching) or across an IP network to another protocol translator (remote switching, also called *tunneling*). Remote X.25 switching encapsulates the X.25 packet-level inside a TCP connection, allowing X.25 equipment to be connected via a TCP/IP-based network.
- Connection-Mode Network Service (CMNS)—CMNS is a mechanism through which local X.25 switching can be extended to different media (Ethernet, FDDI, and Token Ring) by using OSI-based NSAP addresses. This implementation provides X.25 over LLC2 to allow X.25 over media other than serial interfaces. Our CMNS implementation supports services defined in ISO Standards 8208 (packet level) and 8802-2 (frame level).
- DDN and BFE X.25—DDN-specified Standard Service is supported. The DDN X.25 Standard Service is the required protocol for use with DDN packet switched nodes (PSNs). The Defense Communications Agency (DCA) has certified our DDN X.25 Standard Service implementation for attachment to the Defense Data Network. Our DDN implementation also includes Blacker Front-End Encryption and Blacker Emergency Mode software.
- X.25 MIB—Specified in SNMP MIB Extension for X.25 LAPB (RFC 1381) and SNMP MIB Extension for the X.25 Packet Layer (RFC 1382) is supported. However, the LAPB XID Table, X.25 Cleared Circuit Table, and X.25 Call Parameter Table are not implemented. To use the X.25 MIB, refer to your MIB publications.

Our X.25 implementation does not support fast switching.

Support for Terminal Connections and Protocol Translation

The protocol translator uses the CCITT Recommendation X.25 for transferring raw data over X.25 networks. The X.25 software supports both commercial and DDN versions. Protocol translators also support X.25 as a transport mechanism for IP packets, and X.3- and X.29-based PAD connections. This allows the protocol translators to connect to an X.25 public data network (PDN). This X.25 connection allows transport of TCP/IP packets across the X.25 packet-switching network in the same way as would occur on a protocol translator.

Routers do not support an X.25 PAD function, so they are unable to communicate with hosts directly connected to the X.25 PDN. Routers encapsulate TCP/IP packets in X.25 packets for transfer over a packet-switching network. These packets can be received by routers, communication servers, and protocol translators. Only protocol translators include PAD capabilities. However, other servers can communicate with protocol translators using the Telnet or LAT protocols. Protocol translators can translate Telnet and LAT into X.25, which then would be able to communicate with X.25 hosts. Protocol translators support all PAD standards (X.28, X.29, and X.3). The connection to the packet-switching network is through a synchronous line.

Connections to a PAD are made using EXEC commands. You can configure PAD parameter profiles that can be used to set PAD parameters with other commands, and you can configure access lists to control X.25 network access. Both these features use the message fields defined in Recommendation X.29, which describes procedures for exchanging data between two PADs or between a PAD and a DTE.

Facility Handling in Encapsulated X.25 Virtual Circuits

The protocol translator either originates or accepts encapsulation virtual circuits (VCs) in order to transport LAN traffic through an X.25 network.

When the protocol translator originates a CALL for LAN traffic encapsulation, the facilities in the CALL are controlled by the facilities configured for the interface and the map statement that specifies the LAN/X.25 encapsulation. Because a protocol translator might be attached to a public data network (PDN), the interface and map configurations allow a number of facilities to be specified in outgoing CALLs. These facilities are specified in all originated CALLs relating to the given interface and map with one exception: the incoming and outgoing maximum packet sizes proposed will be lowered if the lower layer (LAPB) cannot support the specified data packet size.

When the protocol translator accepts an encapsulation CALL, many facilities are simply ignored. The maximum packet sizes will be lowered if the lower layer (LAPB) cannot support the sizes proposed. A reverse-charged CALL will be cleared if neither the interface nor the map allows it. A CALL that specified a Network User Identification (NUID) will be cleared if the user authentication fails.

Facility Handling in Routed X.25 Virtual Circuits

Routed X.25 traffic may have facilities added, deleted or modified by the protocol translator. The sections that follow discuss standard facilities, CCITT-specified marker facilities, and local marker facilities specified for DDN or BFE X.25.

Standard (1984 X.25) Facilities

The following sections describe how standard (1984 X.25) facilities are treated when routing an switched virtual circuit (SVC).

Flow Control Negotiation (Window Sizes and Maximum Packet Sizes)

The routed X.25 implementation requires that the flow control parameters be negotiated end-to-end and actively intervenes in the negotiation process using the CALL/CALL ACCEPTED packet exchange to ensure this. This behavior is important because routed X.25 traffic is simply passed between the two interfaces (local or remote) without fragmentation, reassembly or local flow control handling. As part of the flow control negotiation process, the following activities take place:

- The protocol translator inserts flow control parameters into the switched CALL if the requested values do not match the outgoing interface's defaults.
- The protocol translator strips flow control parameters from the switched CALL if the requested values match the outgoing interface's defaults.
- Any requested maximum packet size that exceeds the capability of either interface is lowered to the largest value that can be supported.
- Any requested window size that exceeds 7 is lowered to 7 if a CALL is routed from a modulo 128 interface to a modulo 8 interface.
- A remotely routed CALL has both the window sizes and packet sizes inserted into the CALL packet sent over the TCP connection; as noted above, these values are stripped from the outgoing CALL if they match the default values of the outgoing interface. Note that some earlier X.25 implementations did not insert the flow control parameters into the packet; since X.25 cannot determine what the incoming interface's default values are, it indicates the universally acceptable window and packet size values of 2/2 and 128/128 on the CALL ACCEPTED packet.
- An outgoing CALL ACCEPTED packet indicates the accepted flow control values if, for any reason, they are different from the values proposed in the incoming CALL.

Throughput Negotiation

An incoming Throughput facility is forwarded.

Closed User Group Selection

A basic format Closed User Group (CUG) selection facility is forwarded, and any other format of Closed User Group selection (extended format, CUG with outgoing access or Bilateral CUG) is stripped.

Reverse Charging

An incoming Reverse Charging facility is forwarded.

Fast Select

An incoming Fast Select facility is forwarded.

Network User Identification (NUID)

An incoming NUID facility on a CALL packet is forwarded; an NUID facility on a CALL ACCEPTED packet is stripped.

Network User Data Facility (NUDATA)

The NUDATA facility provides a way to specify a user-specified format of the NUID facility field. The NUDATA facility can only be used by X.25 switches that require the NUID facility field to be in a different format from the one provided by the NUID facility. If one of the protocol translators receives a CALL packet that uses the NUDATA format, the CALL is cleared and an error indicating an invalid NUID facility has been specified is sent.

Charging Information

Any Charging Information or Request is stripped.

RPOA Selection

Any RPOA Selection is stripped.

Called Line Address Modified Notification

A Called Line Address Modified notification is forwarded.

Call Redirection Notification

A Call Redirection notification is stripped.

Transit Delay Selection

An incoming Transit Delay facility is forwarded.

CCITT-Specified Marker Facilities

The following sections describe how CCITT-specified marker facilities are treated when routing an SVC.

Calling Address Extension

An incoming Calling Address Extension facility is forwarded.

Called Address Extension

An incoming Called Address Extension facility is forwarded.

Quality of Service Negotiation

Any of the Quality of Service facilities are stripped.

Expedited Data Negotiation

An Expedited Data Negotiation facility is stripped.

Local Marker Facilities Specified for DDN or BFE X.25

The following sections describe how local marker facilities are treated when routing an SVC.

DDN Service Type

An incoming DDN Service Type facility is stripped from a CALL but inserted if a forwarded CALL ACCEPTED packet specifies a DDN precedence facility.

DDN Precedence

An incoming DDN Precedence facility is stripped from a CALL, but forwarded for a CALL ACCEPTED packet.

BFE Emergency Mode Addressing

An incoming BFE Emergency Mode Addressing facility is stripped.

LAPB Configuration Task List

It is possible to use only LAPB as a serial encapsulation method. This can be done using a leased serial line. You must use one of the X.25 packet-level encapsulations when attaching to an X.25 network.

The X.25 standards distinguish between two types of X.25 hosts: DTE and DCE. At Level 2, or the data link layer in the OSI model, X.25 provides LAPB to allow for orderly and reliable exchange of data between a DTE and a DCE. A protocol translator using LAPB encapsulation can act as a DTE or DCE device at the protocol level, which is distinct from the hardware DTE or DCE level.

Using LAPB under noisy conditions can result in greater throughput than HDLC encapsulation. When LAPB detects a missing frame, the protocol translator retransmits the frame instead of waiting for host timers to expire. This behavior is good only if the host timers are relatively slow. In the case

of quickly expiring host timers, however, you will discover that LAPB is spending much of its time transmitting host retransmissions. If the line is not noisy, the lower overhead of HDLC encapsulation is more efficient than LAPB. When using long delay satellite links, for example, the lock-step behavior of LAPB makes HDLC encapsulation the better choice.

Perform the following tasks to configure LAPB:

- Establish LAPB DCE or DTE operation (page 2-6)
- Set the LAPB retransmission timer and frame parameters (page 2-6)
- Define the LAPB hold queue size (page 2-7)

The following sections describe the LAPB configuration tasks. To learn about monitoring and maintaining LAPB, see the section "Monitor and Maintain LAPB and X.25." See the end of the chapter for examples of LAPB configurations.

Establish LAPB DCE or DTE Operation

Set the appropriate LAPB encapsulation to run datagrams over a serial interface. One end of the link must be DTE and the other must be DCE.

Select an encapsulation and the protocol if you are using a single protocol, or select the appropriate multiple protocol operation, according to the following table. Perform these tasks in interface configuration mode.

Task	Command
Enable use of a single protocol on the line using DCE operation.	**encapsulation lapb-dce**
Enable use of a single protocol on the line using DTE operation.	**encapsulation lapb**
Set the line protocol.	**lapb protocol** *protocol*
Enable use of multiple protocols on the line using DCE operation.	**encapsulation multi-lapb-dce**
Enable use of multiple protocols on the line using DTE operation.	**encapsulation multi-lapb**

Set the LAPB Retransmission Timer and Frame Parameters

LAPB operates at the data link layer of the OSI reference model. LAPB specifies methods for exchanging data (in units called *frames*), detecting out-of-sequence or missing frames, retransmitting frames, and acknowledging frames.

When connecting to an X.25 network, use the N1 parameter value set by the network administrator. This value is the maximum size of an X.25 packet. When using LAPB over leased lines, the N1 parameter should be at least eight times the maximum transmission unit (MTU) size. The other frame parameters are determined by the network configuration. See Table 1-1 for the default values.

The retransmission timer determines how long a transmitted frame can remain unacknowledged before the protocol translator polls for an acknowledgment. For X.25 networks, the protocol translator retransmission timer setting should match that of the network.

For leased-line circuits, the retransmission timer setting is critical. The timer setting must be large enough to permit a maximum-sized frame to complete one round trip on the link. If the timer setting is too small, the protocol translator will poll before the acknowledgment frame can return, which results in an effective loss of bandwidth. If the timer setting is too large, the protocol translator waits longer than necessary before requesting an acknowledgment, which also reduces bandwidth.

Note To determine an optimal value for the retransmission timer, use the privileged EXEC command **ping**. For more information on the **ping** command, refer to the **lapb t1** command in the *Protocol Translator Command Reference*.

To set LAPB parameters, perform any of the following tasks in interface configuration mode:

Table 1-1 LAPB Parameters

Task (LAPB Parameter)	Ranges	Default	Command
Set the window size (K).	1-7 packets	7	**lapb k** *window-size*
Set bits per frame (N1).	1088–32840 bits	12056	**lapb n1** *bits*
Set retries for acknowledgment frames (N2).	1-255 retries	20	**lapb n2** *retries*
Set the retransmission timer (T1).	1-64000 milli-seconds	3000	**lapb t1** *milliseconds*

For an example of configuring the LAPB T1 timer, see the section “Typical LAPB Configuration Example” later in this chapter.

Define the LAPB Hold Queue Size

You can define the maximum number of packets to be held while the other side is not accepting data. To do so, perform the following task in interface configuration mode:

Task	Command
Define the size of a packet hold queue.	**lapb hold-queue** *queue-size*

X.25 Configuration Task List

To configure X.25, you can choose from the following tasks, depending upon the X.25 application or task required for your network:

- Configure an X.25 Datagram Transport
- Configure X.25 Routing
- Configure CMNS Routing
- Configure DDN or BFE X.25
- Create X.29 Access Lists
- Create an X.29 Profile Script

- Configure X.25 Level 3 Parameters and Special Features

Note that all X.25 applications except DDN and BFE X.25 require that an encapsulation method and an X.121 address be set; see the sections “Configure X.25 DTE or DCE Operation” and “Set the X.25 Interface Address” for the tasks to do so.

Default X.25 parameters have been provided for X.25 operation; however, you can change the settings to meet the needs of your X.25 network or as defined by your X.25 service supplier. We also provide special and custom configuration settings to further optimize your X.25 network. See the section “Configure X.25 Level 3 Parameters and Special Features” later in this chapter for more information.

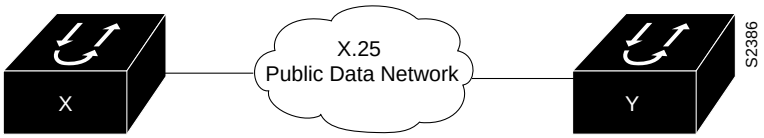
The following sections describe how to implement the supported configuration types and special configuration tasks. See the *Protocol Translator Command Reference* publication for information about the commands in the tasks. To learn about monitoring and maintaining X.25, see the section “Monitoring and Maintaining X.25.” See the end of this chapter for examples of configuring X.25.

Configure an X.25 Datagram Transport

X.25 support is most commonly configured as a transport for datagrams across an X.25 network. This is accomplished by first establishing a mapping between protocol addresses (for example, IP or DECnet) and the X.121 addresses of the X.25 network. When datagrams for a particular destination are routed for the first time, a virtual circuit is set up to the appropriate X.121 address. The Call User Data portion of the initial Call Request identifies the protocol of the datagrams being carried by a particular virtual circuit. If multiple protocols are in use, multiple virtual circuits will be opened.

Figure 1-1 illustrates two protocol translators sending data across an X.25 PDN.

Figure 1-1 Transporting LAN Protocols across an X.25 PDN



You must perform the following tasks to configure your protocol translator as a transport for X.25 datagrams (note that these tasks are required by *all* X.25 applications except DDN and BFE X.25):

- Set X.25 DTE or DCE operation.
- Set the X.25 address.

Perform the following tasks, as necessary, to complete the X.25 configuration for your network needs:

- Map the network address to the X.121 address.
- Establish a permanent virtual circuit (PVC).
- Establish a default VC protocol.

The following sections describe how to perform these configuration tasks. Configuring the X.25 parameters and special features, including TCP header compression and X.25 bridging, is described in the section “Configure X.25 Level 3 Parameters and Special Features” later in this chapter.

Note When configuring IP routing over X.25, you might need to make adjustments to accommodate split horizon effects. Refer to the chapter titled "Configuring IP Routing Protocols" in the *Router Products Configuration Guide* for details about how the protocol translator handles possible split horizon conflicts. By default, split horizon is *enabled* for X.25 networks

Set X.25 DTE or DCE Operation

A protocol translator using X.25 Level 3 encapsulation can act as a DTE or DCE device on general X.25 networks. High-level Data Link Control (HDLC)is the default serial encapsulation method. Configure one of these device types, according to the needs of your X.25 service supplier. To do so, perform one of the following tasks in interface configuration mode:

Task	Command
Set X.25 DTE operation.	**encapsulation x25**
Set X.25 DCE operation.	**encapsulation x25-dce**

Typically a PDN requires attachment as a DTE. (This is distinct from the hardware interface DTE and DCE assignments.)

For an example of configuring X.25 DTE operation, see the section "Typical X.25 Configuration Example" later in this chapter.

Set the X.25 Interface Address

Set the X.121 address of the network interface. The X.121 address is assigned by the X.25 network service provider. To set the X.121 address, perform the following task in interface configuration mode:

Task	Command
Set the X.121 address.	**x25 address** *X.121-address*

For an example of configuring the X.25 interface address, see the section "Typical X.25 Configuration Example" later in this chapter.

Establish an Protocol-to-X.121 Address Map

The protocol translators can use several methods to map protocol addresses to X.121 addresses used by X.25, as follows.

A protocol translator set up for DDN service uses the dynamic mapping technique specified in Appendix A of the *DDN X.25 Host Interface Specification*, available from the DDN Network Information Center (NIC). This technique has two severe limitations: BFE mapping applies only to Class A Internet addresses, and it ignores the third octet of the Internet address. This technique works well for the DDN, but not for other networks. The BFE service is a variant of DDN service and also has a mapping algorithm.

You can establish the X.121 address of a particular network interface using the **x25 address** interface configuration command. If you do not specify the address, the protocol translator uses the DDN mapping technique to obtain the X.121 address of the interface from its IP address.

You also can set up the Internet-to-X.121 address mapping for the host. Because no defined protocol can dynamically determine such mappings, you must enter a mapping for each host with which the protocol translator will exchange traffic.

To establish a dynamic map, perform the following task in interface configuration mode:

Task	Command
Establish an Internet-to-X.121 address map.	**x25 map** *protocol-keyword protocol-address X.121-address* [*options*]

As an example, if you are running IP over X.25, you must define an IP address for the X.25 interface as well as map the IP address to the X.121 address. These tasks are described in the chapter titled "Configuring IP" in the *Router Products Configuration Guide.*

When setting up the address map, you can include features such as directed broadcasts, virtual circuit limits, and user facility settings as part of the options. The following list summarizes the supported protocols for map definitions:

- IP
- DECnet
- XNS
- Novell IPX
- AppleTalk
- VINES
- Apollo Domain
- Bridging
- OSI Connectionless Network Service
- OSI Connection-Mode Network Service
- TCP header compression

For specific information about how to establish a protocol to run over X.25, refer to each of the protocol chapters in the *Router Products Configuration Guide.*

The configuration for the Open Shortest Path First (OSPF) protocol can be greatly simplified by adding the optional **broadcast** keyword when doing this task. See the **x25-map** description in the "X.25 Configuration Commands" chapter of the *Protocol Translator Command Reference* for more information.

Establish a Permanent Virtual Circuit (PVC)

Permanent virtual circuits (PVCs) are the X.25 equivalent of leased lines; they are never disconnected. Specify the required network protocol-to-X.121 address map before you set up a PVC. To establish a PVC, perform the following task in interface configuration mode:

Task	Command
Set a PVC.	**x25 pvc** *circuit protocol-keyword protocol-address* [*option*]

See the previous section for a list of protocols; all but OSI and TCP header compression are supported by this task.

Establish a Default VC Protocol

The Call Request packet that sets up a VC might contain a field called the Call User Data (CUD) field. Typically, the software uses the first byte of CUD to distinguish which high-level protocol will be carried by a particular virtual circuit.

Table 1-2 lists the hexadecimal values of the initial byte of CUD and its corresponding network level protocol. The use of 0x81 for ISO 8473 (CLNS) is an ISO standard. The use of 0xCC for Department of Defense IP is defined by RFC 877. The use of C0 00 80 C4 is defined by Banyan. The other values are meaningful only to the X.25 software. Most of the single-byte identifiers are padded to four bytes with three bytes of 0x00. BFE IP encapsulation requires that only one byte be used.

Table 1-2 Protocols and Initial Byte of Call User Data

Protocol	Initial CUD Byte
DOD IP	0xCC
Novell IPX	0xD3
TCP Header Compression (THC)	0xD8

You can specify the protocol assumed by the protocol translator to interpret calls with unknown CUD or with no CUD. The protocol translator supports the IP protocol and X.3 PAD connections. To do this, perform the following task in interface configuration mode:

Task	Command
Establish the default protocol.	**x25 default** *protocol*

Configure X.25 Routing

The X.25 software implementation allows virtual circuits to be routed from one X.25 interface to another and from one protocol translator to another. The routing behavior can be controlled with switching and tunneling commands, based on a locally built table.

Higher-level protocols can share an X.25 encapsulated serial interface with the X.25 switching support. Switching or forwarding X.25 virtual circuits can be done two ways:

- Incoming calls received from a local serial interface running X.25 can be forwarded to another local serial interface running X.25. This is known as *local X.25 switching* because the protocol translator handles the complete path. It does not matter whether the interfaces are configured as DTE or DCE devices; software will take the appropriate actions.
- An incoming call also can be forwarded to another Cisco protocol translator over a LAN using the TCP/IP protocols. Upon receipt of an incoming call, a TCP stream connection is established to the protocol translator that is acting as the switch for the destination. All X.25 packets are sent and received over this reliable data stream. Flow control is maintained from local interface to remote interface. This is known as *remote X.25 switching*, or *tunneling.* It does not matter whether the interfaces are configured as DTE or DCE, because software will take the appropriate actions.

Running X.25 over TCP/IP provides a number of benefits. The datagram containing the X.25 packet can be switched by other protocol translators using their high-speed switching abilities. X.25 connections can be sent over networks running only the TCP/IP protocols. The TCP/IP protocol suite runs over many different networking technologies, including Ethernet, Token Ring, T1 serial, and FDDI. Thus X.25 data can be forwarded over these media to another protocol translator, where it can be output to an X.25 interface.

When the connection is made locally, the switching configuration is used; when the connection is across a LAN, the tunneling configuration is used. The basic function is the same for both types of connections, but different configuration commands are required for the two types of connections.

The X.25 switching subsystem supports the following facilities and parameters:

- The D-bit ignored but passed through transparently
- Variable-length interrupt data
- Flow Control Parameter Negotiation
  - Window size up to 7, or 127 for modulo 128 operation
  - Packet size up to 4096
- The basic Closed User Group selection
- Throughput class negotiation
- Reverse charging and fast select

The tasks for configuring these facilities are described in the section “Configure X.25 Level 3 Parameters and Special Features.”

Perform the following tasks to configure X.25 tunneling on your protocol translator:

- Enable local X.25 routing.
- Construct the X.25 routing table.
- Translate X.25 called and calling addresses, if needed.
- Update the X.121 address.
- Configure a switched PVC.
- Configure a tunneled PVC.

You also need to configure the X.25 parameters and special features, as required for your network. Each task is described in a following section.

Enable X.25 Routing

You can enable local switching or tunneling (used for remote switching), allowing you to route X.25 traffic through a LAN. To enable local X.25 routing, perform the following global configuration task:

Task	Command
Enable X.25 routing.	**x25 routing**

Construct the X.25 Routing Table

An X.25 routing table is consulted when an incoming call is received that cannot be a datagram transport virtual circuit. Two fields are used to determine the route: the called X.121 network interface address (or destination host address) and the X.25 packet's CUD field. When the destination address and the CUD of the incoming packet fit the X.121 and CUD patterns in the routing table, the packet is forwarded. Additionally, you can translate called and calling addresses. To construct the X.25 routing table, you must perform one of the following tasks in global configuration mode:

Task	Command
Construct the X.25 routing table by interface type.	**x25 route** [*# position*] *X.121-pattern* [**cud** *pattern*] **interface** *interface number*
Construct the X.25 routing table by IP address (tunneling).	**x25 route** [*# position*] *X.121-pattern* [**cud** *pattern*] **ip** *IP-address* [*IP-address2...IP-address6*]

Entries have as a destination an interface (local) or IP address (remote). Up to six remote IP addresses can be entered as alternates routes that will be tried in turn until a successful connection is made.

Note It can take up to 50 seconds to try an alternate IP address due to TCP timings.

Translate X.25 Called and Calling Addresses

When interconnecting two separate X.25 networks, you must sometimes provide for address translation. Your X.25 switch supports translation of X.25 called and calling addresses.

To translate X.25 called and calling addresses, perform the following global configuration task:

Task	Command
Translate X.25 called and calling addresses.	**x25 route** [*# position*] *X.121-pattern* [**substitute-source** *rewrite-pattern*] [**substitute-dest** *rewrite-pattern*] [**cud** *pattern*] **interface** *interface number*

Note that address substitution is only performed on routes to an interface. When running X.25 over IP, address substitution can be performed on the destination IP system if the destination system is configured with the appropriate X.25 routing commands.

Update the X.121 Address

Some X.25 calls, when forwarded by the X.25 switching support, need the calling (source) X.121 address updated to that of the outgoing interface. This is necessary when forwarding calls from private data networks to public data networks; other networks might require either address to be suppressed.

To update or suppress the X.121 address, perform one of the following tasks in interface configuration mode:

Task	Command
Update the X.121 address.	**x25 use-source-address**
Suppress the calling (source) X.121 address in Call Request packets.	**x25 suppress-calling-address**
Omit the called (destination) X.121 address in Call Request packets.	**x25 suppress-called-address**

Configure a Switched PVC

You can configure an X.25 PVC in the X.25 switching software. This means that DTEs that require permanent circuits can be connected to a protocol translator acting as an X.25 switch and have a properly functioning connection. X.25 RESETs will be sent to indicate when the circuit comes up or goes down. Both interfaces must define complementary switched PVCs.

To configure a switched PVC, perform the following task in interface configuration mode:

Task	Command
Configure a switched PVC.	**x25 pvc** *pvc-number1* **interface** *interface-name* **pvc** *pvc-number2* [*option*]

For an example of configuring a local switched PVC, see the section “Remote Tunneling Example” later in this chapter

Configure a Tunneled PVC

A PVC can be forwarded to another protocol translator over a LAN using the TCP/IP protocols. When the interfaces come up, a TCP stream connection is established to the protocol translator that is acting as the switch for the destination. All X.25 packets will be sent and received over this reliable data stream. Flow control is maintained from local DTE to remote DTE. This is known as *remote PVC switching* or *tunneling.*

Running X.2 over TCP/IP provides a number of benefits. Other protocol translators can switch IP datagrams containing the X.25 packets using the protocol translator’s high-speed switching abilities. X.25 data can be sent over networks running only TCP/IP protocols. The TCP/IP protocol suite runs

over many different networking technologies, including Ethernet, Token Ring, T1 serial, and FDDI. Thus X.25 data can be forwarded over these media to another protocol translator where it can be output to an X.25 interface. Both interfaces must define complementary tunneled PVCs.

To connect two PVCs across a TCP/IP LAN, perform the following task in interface configuration mode:

Task	Command
Connect two PVCs across a TCP/IP LAN.	**x25 pvc** *pvc-number1* **tunnel** *IP-address* **interface serial** *string* **pvc** *pvc-number2* [*options*]

See the section "LAPB and X.25 Configuration Examples" later in this chapter for an example of how to configure this feature.

Configure CMNS Routing

The Connection-Mode Network Service (CMNS) provides a mechanism through which local X.25 switching can be extended to nonserial media by using OSI-based NSAP addresses. This implementation runs packet-level X.25 over frame-level LLC2.

In addition, the CMNS implementation allows LAN-based OSI resources, such as a DTE host and a Sun workstation, to be interconnected to each other via the protocol translator's LAN interfaces and to a remote OSI-based DTE through a WAN interface (using, for example, an X.25 PSN).

Note CMNS is implicitly enabled whenever an X.25 encapsulation is included with a serial interface configuration.

All local mapping is performed by statically mapping MAC addresses to NSAP addresses and X.121 addresses to NSAP addresses.

Implementing CMNS routing involves the following basic steps:

- Enable CMNS on an interface.
- Specify a CMNS static map of addresses.

See the following sections for more specific information about these tasks.

Enable CMNS on an Interface

Enable CMNS on a nonserial interface by performing the following task in interface configuration mode:

Task	Command
Enable CMNS.	**cmns enable**

For an example of enabling CMNS on an interface, see the section "Enabling CMNS for X.121 and MAC Addresses Example" later in this chapter.

Specify a CMNS Static Map of Addresses

After enabling CMNS on a nonserial interface (or specifying X.25 encapsulation on a serial interface), you must map NSAP addresses to either MAC-layer addresses or X.121 addresses, depending on the application.

For CMNS support over dedicated serial links (such as leased lines), an X.121 address is not needed, but can be included. You must specify the X.121 address for CMNS connections over a packet-switched network, and you must specify a MAC address for CMNS connections over a nonserial media (Ethernet, FDDI, or Token Ring).

Map the NSAP addresses to either a MAC address or X.121 address by performing one of the following tasks in interface configuration mode:

Task	Command
Statically map NSAP addresses to MAC-layer address.	**x25 map cmns** *NSAP MAC-address*
Statically map NSAP addresses to X.121 address.	**x25 map cmns** *NSAP* [*X.121-address*]

For an example of configuring a CMNS static map of addresses, see the section "Example of Enabling CMNS for X.121 and MAC Addresses" later in this chapter.

Configure DDN or BFE X.25

The DDN X.25 protocol has two versions: Basic Service and Standard Service. Our X.25 implementation only supports the Standard Service. DDN X.25 Standard Service requires that the X.25 data packets carry IP datagrams. The DDN packet switch nodes (PSNs) can extract the IP packet from within the X.25 packet and pass data to another Standard Service host.

The DDN X.25 Standard is the required protocol for use with DDN PSNs. The Defense Communications Agency (DCA) has certified our DDN X.25 Standard implementation for attachment to the DDN. As part of the certification, our software is required to provide a scheme for dynamically mapping Internet addresses to X.121 addresses. See the section "DDN X.25 Dynamic Mapping" for details on that scheme.

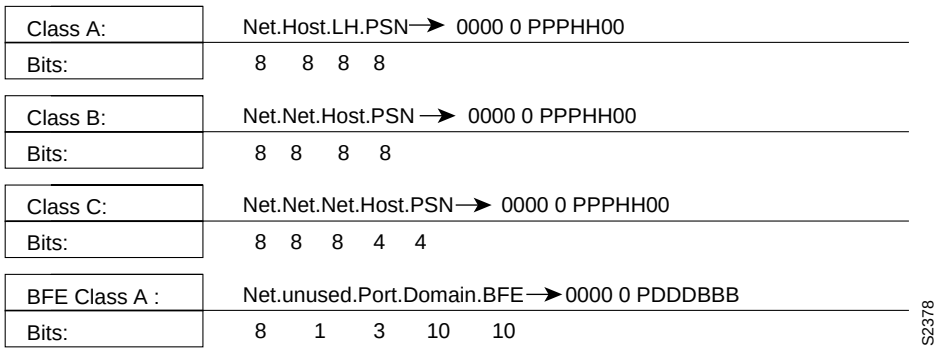
Complete the following tasks to enable DDN X.25 service:

- Enable DDN X.25
- Define the IP precedence handling
- Configure Defense Data Network Blacker Front-End Encryption (BFE) and Blacker Emergency Mode, as needed.

DDN X.25 Dynamic Mapping

The DDN X.25 standard implementation includes a scheme for dynamically mapping all classes of Internet addresses to X.121 addresses without a table. This scheme requires that the IP addresses conform to the formats shown in Figure 1-2. These formats segment the Internet addresses into network (N), host (H), logical address (L), and PSN (P) portions. (PSN is an acronym for Packet Switch Node.) For the BFE encapsulation, the Internet address is segmented into Port (P), Domain (D), and BFE ID number (B). The DDN algorithm requires that the host value be less than 64.

Figure 1-2 DDN Internet/X.121 Address Conventions



The DDN conversion scheme uses the host and PSN portions of an Internet address to create the corresponding X.121 address. Strictly speaking, the DDN conversion mechanism is limited to Class A Internet addresses. However, the protocol translator can convert Class B and Class C addresses as well. As indicated, this method uses the last two octets of a Class B address as the host and PSN identifiers, and the upper and lower four bits in the last octet of a Class C address as the host and PSN identifiers, respectively. The BFE conversion scheme requires a Class A IP address.

The DDN conversion scheme uses a physical address mapping if the host identifier is numerically less than 64. (This limit derives from the fact that a PSN cannot support more than 64 nodes.) If the host identifier is numerically larger than 64, the resulting X.121 address is called a logical address. The DDN does not use logical addresses.

The format of physical DDN X.25/X.121 addresses is ZZZZFIIIHHZZ(SS), where each character represents a digit. ZZZZ represents four zeros, F is zero to indicate a physical address, III represents the PSN octet from the Internet address padded with leading zeros, HH is the host octet from the Internet address padded with leading zeros, and ZZ represents two zeros. (SS) represents the optional and unused subaddress.

The physical and logical mappings of the DDN conversion scheme always generate a12-digit X.121 address. Subaddresses are optional; when added to this scheme, the result is a 14-digit X.121 address. The DDN does not use subaddressing.

Packets using routing and other protocols that require broadcast support can successfully traverse X.25 networks, including the DDN. This traversal requires the use of network protocol-to-X.121 maps, because the protocol translator must know explicitly where to deliver broadcast datagrams. (X.25 does not support broadcasts.) You can mark network protocol-to-X.121 map entries to accept broadcast packets; the protocol translator then sends broadcast packets to hosts with marked entries. If you do not specify the address for an interface configured for DDN X.25, the protocol translator uses the DDN mapping technique to obtain the X.121 address of an interface.

Enable DDN X.25

Both DCE and DTE operation cause the protocol translator to specify the Standard Service facility in the Call Request packet, which notifies the PSNs to use Standard Service.

Perform one of the following tasks in interface configuration mode, as appropriate for your network:

Task	Command
Set DDN X.25 DTE operation.	**encapsulation ddnx25**
Set DDN X.25 DCE operation.	**encapsulation ddnx25-dce**

For an example of enabling DDN X.25, see the section "DDN X.25 Configuration Example" later in this chapter.

Define IP Precedence Handling

Using Standard Service, the DDN can be configured to provide efficient service for datagrams with high precedence values. If the protocol translator receives an Internet packet with a nonzero Internet precedence field, it opens a new virtual circuit and sets the precedence facility request to the DDN-specified precedence mapping in the Call Request packet. Different virtual circuits are maintained based on the precedence mapping values and the number of virtual circuits permitted.

The DDN X.25 software opens one virtual circuit for all types of service values.

To enable the precedence handling feature, perform the following task in interface configuration mode:

Task	Command
Allow a new virtual circuit based on the type of service (TOS) field.	**x25 ip-precedence**

Some hosts send nonstandard data in the TOS field, thereby causing multiple, wasteful virtual circuits to be created.

Configure Blacker Emergency Mode

For environments that require a high level of security, your protocol translator software supports the DDN Blacker Front-End Encryption (BFE) and Blacker Emergency Mode.

Blacker Emergency Mode allows your BFE device and your protocol translator to function in emergency situations. When the protocol translator is configured to participate in emergency mode and the BFE device is in emergency mode, the protocol translator sends address translation information to the BFE device to assist it in sending information.

Our implementation of Blacker Emergency Mode adheres to the specifications outlined in the DCA Blacker Interface Control document, published March 21, 1989.

Your BFE device is configured to be in one of three possible modes as follows:

- Enters emergency mode when requested to by the network. If the protocol translator is configured to respond to a BFE device in emergency mode, or if the EXEC command **bfe enter** is used, the protocol translator sends address translation information to the BFE device.

- Never enters emergency mode.
- Notifies the protocol translator that an emergency mode window is open and waits for the protocol translator to tell it to enter emergency mode. If the protocol translator is configured to respond to a BFE in emergency mode, or if the EXEC command **bfe enter** is used, the protocol translator sends a special address translation packet to the BFE device. The "special" data includes a command to the BFE to enter emergency mode.

Perform these tasks to configure Blacker Emergency Mode:

- Set BFE encapsulation on the protocol translator attached to a BFE device.
- Provide address translation information to the BFE device.
- Define the circumstances under which the protocol translator will participate in emergency mode.
- Enter Blacker Emergency Mode using the **bfe** EXEC command.

The following tables describe these tasks.

BFE encapsulation operates to map between Class A IP addresses and the X.121 addresses expected by the BFE encryption device. To set BFE encapsulation, perform the following task in interface configuration mode:

Task	Command
Set BFE encapsulation on the protocol translator attached to a BFE device.	**encapsulation bfex25**

For an example of enabling BFE Emergency mode, see the section "BFE Emergency Mode Example" later in this chapter.

You must set up a table that provides the address translation information the protocol translator sends to the BFE when the BFE is in emergency mode. To do so, perform the following task in interface configuration mode:

Task	Command
Set up the table that lists the BFE nodes (host or gateways) to which the protocol translator will send packets.	**x25 remote-red** *host-IP-address* **remote-black** *Blacker-IP-address*

For an example of configuring setting up the table, see the section "BFE Emergency Mode Example" later in this chapter.

You can define the circumstances under which the protocol translator participates in emergency mode and how it will participate in emergency mode. To do so, perform the following tasks in interface configuration mode:

Task	Command
Configure the circumstances under which the protocol translator will participate in emergency mode.	**x25 bfe-emergency** {**never** | **always** | **decision**}
Configure how a protocol translator configured as **x25 bfe-emergency decision** will participate in emergency mode.	**x25 bfe-decision** {**no** | **yes** | **ask**}

For an example of configuring the circumstances under which the protocol translator participates in emergency mode, see the section "BFE Emergency Mode Example" later in this chapter.

To set the protocol translator to participate in emergency mode or to end participation in emergency mode when your system is so configured, perform the following task in EXEC mode:

Task	Command
Set the protocol translator to participate in emergency mode.	**bfe** {**enter** | **leave**} *interface-type number*

For an example of configuring the protocol translator to participate in emergency mode, see the section "BFE Emergency Mode Example" later in this chapter. See the previous task for more information about configuring the protocol translator to participate in emergency mode.

Create X.29 Access Lists

The protocol translator software provides access lists, which make it possible to limit access to the protocol translator from certain X.25 hosts. Access lists take advantage of the message field defined by Recommendation X.29, which describes procedures for exchanging data between two PADs or between a PAD and a DTE device.

To define X.29 access lists, perform the following tasks:

Step 1 Create an access list.

Step 2 Apply an access list to a virtual line or to a translate command.

These tasks are described in the following sections.

When configuring protocol translation, you can specify an access list number with each **translate** command. In the case of translation sessions that result from incoming PAD connections, the corresponding X.29 access list is used.

Create an Access List

To specify the access conditions, perform the following global configuration task:

Task	Command
Restrict incoming and outgoing connections between a particular virtual terminal line (into a Cisco device) and the addresses in an access list.	**X29 access-list-number** {**permit**|**deny**} *regular-expression*

An access list can contain any number of lines. The lists are processed in the order in which you type the entries. The first match causes the permit or deny condition. If an X.121 address does not match any of the entries in the access list, access will be denied.

Apply an Access List to a Virtual Line

To apply an access list to a virtual line, perform the following tasks:

Task		Command
Step 1	Specify a virtual line.	**line vty** *x*
Step 2	Restrict incoming and outgoing connections between a particular virtual terminal line (into a Cisco device) and the addresses in an access list.	**access-class** *access-list-number* **in**

The access list number is used both for incoming TCP access and for incoming PAD access. For TCP access, the protocol translator uses the defined IP access lists. For incoming PAD connections, the same X.25 access list is used. If you want to have access restrictions only on one of the protocols, then you can create an access list that permits all addresses for the other protocol.

Note For an example of applying an access list to a translate command, see the section "Example of an X.29 Access List." at the end of this chapter.

Create an X.29 Profile Script

You can create an X.29 profile script for use by the **translate** command. When an X.25 connection is established, the protocol translator then acts as if an X.29 SET PARAMETER packet had been sent that contained the parameters and values set by this command.

To create an X.29 profile script, perform the following global configuration task:

Task	Command
Create an X.29 profile script.	**x29 profile** *name parameter*:*value* **[***parameter*:*value***...]**

Configure X.25 Level 3 Parameters and Special Features

The protocol translator software allows you to configure the standard Level 2 and Level 3 X.25 parameters and user facilities.

Note If you connect a protocol translator to an X.25 network, use the parameters set by the network administrator. Also, note that the X.25 Level 2 parameters described in "Set the LAPB Retransmission Timer and Frame Parameters" earlier in this chapter affect X.25 Level 3 operations.

This section describes the X.25 parameters, user facilities, and special features you can configure. Which tasks you perform depends upon the structure of your network and the requirements of the service provider. These parameters must be adjusted to match the values used by the X.25 network. It is common for networks to require values different from the defaults.

Perform the following tasks to configure the optional parameters, user facilities, and special features:

- Configure virtual circuits ranges.
- Clear the switched virtual circuit idle timer.
- Increase the number of virtual circuits allowed.
- Configure the ignore destination timer.
- Set default window sizes.
- Set default packet sizes.
- Establish the Level 3 packet acknowledgment policy.
- Configure the Level 3 retransmission timers.
- Set X.25 TCP header compression.
- Define Call User Data.
- Configure the essential X.25 user facilities.
- Define the extended packet sequence.
- Suppress the calling address.
- Suppress the called address.
- Control packet-level restarts.
- Define the packet hold queue size.
- Configure an interface alias.
- Enable X.25 bridging.

The following sections describe these tasks.

Configure Virtual Circuit Ranges

The X.25 protocol maintains multiple connections over one physical link between a DTE and a DCE. These connections are called virtual circuits (VCs) or logical channels (LCs). X.25 can maintain up to 4095 VCs numbered 1 through 4095. An individual VC is identified by giving its logical channel identifier (LCI) or virtual circuit number (VCN). Many documents use the terms VC and LC, and VCN, LCN, and LCI interchangeably. Each of these terms refer to the virtual circuit number.

An important part of X.25 operation is the range of virtual circuit numbers. Virtual circuit numbers are broken into four ranges (listed here in numerically increasing order):

1. Permanent virtual circuits (PVCs)
2. Incoming-only circuits
3. Two-way circuits
4. Outgoing-only circuits

The incoming-only, two-way, and outgoing-only ranges define the VC numbers over which a switched virtual circuit (SVC) can be established by placing an X.25 call, much like a telephone network establishes a switched voice circuit when a call is placed.

The rules about DCE and DTE devices initiating calls are as follows:

- Only the DCE device can initiate a call in the incoming-only range.

- Only the DTE device can initiate a call in the outgoing-only range.
- Both the DCE device and the DTE device can initiate a call in the two-way range.

There is no difference in the operation of the SVCs except the restrictions on which a device can initiate a call. These ranges can be used to prevent one side from monopolizing the virtual circuits, which can be useful for X.25 interfaces with a small total number of SVCs available.

Six X.25 parameters define the upper and lower limit of each of the three SVC ranges. A PVC must be assigned a number less than the numbers assigned to the SVC ranges. An SVC range is not allowed to overlap another range.

Table 1-3 lists the virtual circuit range types and the corresponding commands used to configure the ranges. Note that the values for these parameters must be the same on both ends of an X.25 link. For connection to a public data network (PDN), these values must be set to the values assigned by the network. An SVC range is unused if its lower and upper limits are set to 0; other than this use for marking unused ranges, VC 0 is not available.

Note Because the X.25 protocol requires the DTE and DCE to have identical VC ranges, if the interface is up, changes to the VC range limits will be held until the X.25 protocol RESTARTs the packet service.

Table 1-3 X.25 Virtual Circuits

Task (Parameter)	Range	Default	Command
Set the lowest incoming-only virtual circuit number (LIC).	1-4095	0	**x25 lic** *circuit-number*
Set the highest incoming-only virtual circuit number (HIC).	1-4095	0	**x25 hic** *circuit-number*
Set the lowest two-way virtual circuit number (LTC).	1-4095	1	**x25 ltc** *circuit-number*
Set the highest two-way virtual circuit number (HTC).	1-4095	1024	**x25 htc** *circuit-number*
Set the lowest outgoing-only virtual circuit number (LOC).	1-4095	0	**x25 loc** *circuit-number*
Set the highest outgoing-only virtual circuit number (HOC).	1-4095	0	**x25 hoc** *circuit-number*

Clear the Switched Virtual Circuit Idle Timer

The protocol translator can clear a switched virtual circuit (SVC) after a set period of inactivity. You can set this time by performing the following task in interface configuration mode:

Task	Command
Set the idle time before an SVC is cleared.	**x25 idle** *minutes*

See the section “Monitor and Maintain LAPB and X.25” later in this chapter for additional commands that clear virtual circuits.

Increase the Number of Virtual Circuits Allowed

For X.25 datagram transport, you can establish up to eight switched virtual circuits to a host for each protocol. Perform the following tasks to increase the number of VCs allowed:

Task	Command
Specify the maximum number of switched virtual circuits that can be open simultaneously to one host for each protocol.	**x25 nvc** *count*
Map an X.121 address to the protocol.	**x25 map** *protocol-keyword protocol-address X.121-address* **nvc** *count*

Configure the Ignore Destination Timer

Upon receiving a Clear Request for an outstanding Call Request, the X.25 support code immediately tries another Call Request if it has more traffic to send. This can overrun some X.25 switches. You can define the number of minutes it takes to prevent calls from going to a previously failed destination. Incoming calls will still be accepted.

To configure the ignore destination timer, perform the following task in interface configuration mode:

Task	Command
Configure the ignore destination timer.	**x25 hold-vc-timer** *minutes*

Set Default Window Sizes

X.25 networks have a default maximum input and output window size that can be set by the network administrator. You can set the protocol translator input and output window sizes to match those of the network; see the note in the next section, “Set Default Packet Sizes.”

To configure the default window size, perform the following tasks in interface configuration mode:

Task	Command
Determine how many packets the server can receive before sending an X.25 acknowledgment.	**x25 win** *packets*
Determine how many sent packets can remain unacknowledged before the server uses a hold queue.	**x25 wout** *packets*

Set Default Packet Sizes

X.25 networks have a default maximum input and output packet size that can be set by the network administrator. You can set the protocol translator input and output packet sizes to match those of the network.

To set the default packet size, perform the following tasks in interface configuration mode:

Task	Command
Set the input packet size.	**x25 ips** *bytes*
Set the output packet size.	**x25 ops** *bytes*

To send a packet larger than the X.25 packet size over an X.25 virtual circuit, a protocol translator must break the packet into two or more X.25 packets with the M-bit ("more data" bit) set. The receiving device collects all packets with the M-bit set and reassembles them.

Note Because the X.25 protocol requires the DTE and DCE to have identical default packet and window sizes, if the interface is up, changes to the window and packet sizes will be held until the X.25 protocol RESTARTs the packet service.

Establish the Packet Acknowledgment Policy

You can instruct the protocol translator to send acknowledgment packets when it is not busy sending other packets, even if the number of input packets has not reached the input window count. This approach improves line responsiveness at the expense of bandwidth.

To establish the packet acknowledgment policy, perform the following step in interface configuration mode:

Task	Command
Instruct the server to delay acknowledgment of the data packets.	**x25 th** *delay-count*

Configure the X.25 Level 3 Retransmission Timers

The X.25 Level 3 retransmission timers determine how long the protocol translator must wait before retransmitting various packets. You can set these timers independently using the commands listed in Table 1-4. Each keyword requires a time value in seconds as its argument. The last column shows the default timer values, in seconds. Four of the timers apply to DTE devices, and the other four apply to DCE devices.

Table 1-4 X.25 Retransmission Timers

Task	Default Timer Value	Command
Set DTE T20 Restart Request.	180 seconds	**x25 t20** *seconds*
Set DCE T10 Restart Request.	60 seconds	**x25 t10** *seconds*
Set DTE T21 Call Request.	200 seconds	**x25 t21** *seconds*
Set DCE T11 Call Request.	180 seconds	**x25 t11** *seconds*
Set DTE T22 Reset Request.	180 seconds	**x25 t22** *seconds*
Set DCE T12 Reset Request.	60 seconds	**x25 t12** *seconds*
Set DTE T23 Clear Request.	180 seconds	**x25 t23** *seconds*
Set DCE T13 Clear Request.	60 seconds	**x25 t13** *seconds*

Set X.25 TCP Header Compression

The protocol translator software supports RFC 1144 TCP/IP header compression on serial lines using HDLC and X.25 encapsulation. The implementation of Compressed TCP over X.25 uses one VC to pass the compressed packets. The noncompressed packets are carried over separate IP encapsulations.

To set the X.25 TCP header compression, perform the following tasks in interface configuration mode:

Task	Command
Implement header compression on IP packets.	**ip tcp header-compression** [**passive**]
Allow a separate VC for compressed packets.	**x25 map compressedtcp** *IP-address X.121-address* [*options*]

Define Call User Data

Part of the X.25 Call packet structure includes a *Call User Data* field, a final optional field that enables users to transparently pass small amounts of user data to the called DTE.

The following is an overview of the X.25 commands that support use of the Call User Data field.

Task		Command
Step 1	Specify four bytes of data in the optional **cud** *pattern* argument by interface type,	**x25 route** [*# position*]*X.121-pattern* [**cud** *call-user-data*] **interface** *interface-name*
	or	
	Specify the four bytes of data by IP address.	**x25 route** [*# position*]*X.121-pattern* [**cud** *call-user-data*] **ip** *IP-address*
Step 2	Set the Call User Data field in the X.25 Call Request packet.	**x25 host** *name X.121-address* [**cud** *call-user-data*]
Step 3	Interpret incoming calls with unknown Call User Data.	**x25 default** *protocol*

Configure the X.25 User Facilities

The X.25 software provides commands to support the X.25 user facilities—options specified by the creators of the X.25 Recommendation that allow the network planner to implement features such as accounting, user identification, and flow control negotiation. Table 1-5 lists the facilities supported and the commands that configure them. The facilities configured by the **x25 map** commands are on a per-peer basis; all other facility commands specify the values sent for calls originated by the interface.

Table 1-5 Supported X.25 User Facilities

Task (User Facility)	Command
Select closed user group.	**x25 facility cug** *number* **x25 map** *protocol-keyword protocol-address X.121-address* **cug** *number*
Set flow control parameter negotiation values.	**x25 facility packetsize** *in-size out-size* **x25 map** *protocol-keyword protocol-address X.121-address* **packetsize** *in-size out-size* **x25 facility windowsize** *in-size out-size* **x25 map** *protocol-keyword protocol-address X.121-address* **windowsize** *in-size out-size*
Set reverse charging.	**x25 facility reverse** **x25 map** *protocol-keyword protocol-address X.121-address* **reverse**
Allow reverse charging acceptance.	**x25 accept-reverse** **x25 map** *protocol-keyword protocol-address X.121-address* **accept-reverse**
Select throughput class negotiation.	**x25 facility throughput** *in out* **x25 map** *protocol-keyword protocol-address X.121-address* **throughput** *in out*
Select transit delay.	**x25 facility transit-delay** *number* **x25 map** *protocol-keyword protocol-address X.121-address* **transit-delay** *number*
Set the Recognized Private Operation Agency (RPOA) definition to use.	**x25 facility rpoa** *name* **x25 rpoa** *name number ...*
Set the Cisco standard network user identification.	**x25 map** *protocol-keyword protocol-address X.121-address* **nuid** *username password*
Set the CCITT-type network user identification allowing format determined by the network administrator.	**x25 map** *protocol-keyword protocol-address X.121-address* **nudata** *string*

Additionally, the D-bit is supported and passed through transparently. Both restricted and unrestricted fast select are also supported and are transparently handled by the software. No configuration is required for use of the D-bit or fast select facilities.

Define the Extended Packet Sequence

Our implementation of X.25 supports the extended packet sequence numbering. To define the extended packet sequence, perform the following task in interface configuration mode:

Task	Command
Define the extended packet sequence.	**x25 modulo** *modulus*

Note Because the X.25 protocol requires the DTE and DCE to have identical default packet sizes, if the interface is up, changes to the packet sizes will be held until the X.25 protocol RESTARTs the packet service.

Suppress the Calling Address

You can omit the calling (source) X.121 address in outgoing calls. This option is required for networks that expect only subaddresses in the calling address field.

To suppress the calling address, perform the following task in interface configuration mode:

Task	Command
Omit the calling (source) X.121 address in Call Request packets.	**x25 suppress-calling-address**

Suppress the Called Address

You can omit the called (destination) X.121 address in outgoing calls. This option is required for networks that expect only subaddresses in the called address field.

To suppress the called address, perform the following task in interface configuration mode:

Task	Command
Omit the called (destination) X.121 address in Call Request packets.	**x25 suppress-called-address**

Control Packet-Level Restarts

By default, a packet-level Restart request is forced when the link level is restarted. This behavior can be disabled for networks that do not allow it, but disabling this behavior can cause anomalous packet layer behavior.

To disable packet-level restarts, perform the following task in interface configuration mode:

Task	Command
Disable packet-level restarts.	**no x25 linkrestart**

Define the Packet Hold Queue Size

To define the maximum number of packets that can be held while a virtual circuit is unable to send data, perform the following task in interface configuration mode:

Task	Command
Define the size of a packet hold queue.	**x25 hold-queue** *queue-size*

Configure an Interface Alias

You can supply alias X.121 addresses for an interface. This allows the interface to accept data encapsulation calls to an address other than the address defined by the **x25 address** command.

To configure an interface alias, perform the following global configuration task:

Task	Command
Supply an alias for other X.121 addresses that will accept calls for the protocol translator.	**x25 route** [*# position*] *X.121-pattern* [**cud** *pattern*] **alias** *interface number*

Monitor and Maintain LAPB and X.25

To monitor and maintain the X.25 and LAPB connections, perform one or more of the following tasks:

Task	Command
Clear all virtual circuits at once (calls originated, terminated, and switched by the protocol translator are cleared), or clear the single VC specified.	**clear x25-vc** *interface number* [*lcn*]
Display CMNS information.	**show cmns** *[interface-name]*
Display operation statistics for an interface.	**show interfaces serial** *number*
Display CMNS connections over LLC2.	**show llc2**
Display the protocol-to-X.121 address map.	**show x25 map**
Display the one-to-one mapping of the host IP addresses and the remote BFE device's IP addresses.	**show x25 remote-red**
Display routes assigned by the **x25 route** command.	**show x25 route**
Display details of active VCs.	**show x25 vc** [*lcn*]

Note Refer to the *Debug Command Reference* publication for a description of X.25 diagnostic codes that can appear in these **show** command displays.

LAPB and X.25 Configuration Examples

Use the examples in this section to help you understand how to configure LAPB and X.25 for your network. The following examples are provided:

Example of a Typical LAPB Configuration

In the following example, the frame size (N1), window size (K), hold timer (TH), and maximum retransmission (N2) parameters retain their default values. The **encapsulation** configuration command sets DCE operation for IP packets only, and the **lapb t1** configuration command sets the retransmission timer to 4,000 milliseconds (4 seconds) for a link with a long delay or slow connecting DTE device.

```
interface serial 3
encapsulation lapb-dce
lapb t1 4000
```

Example of a Typical X.25 Configuration

The following example shows the complete configuration for a serial interface connected to a commercial X.25 PDN for routing the IP protocol. The IP subnetwork address 131.108.9.0 has been assigned for the X.25 network.

Note When routing IP over X.25, the X.25 network must be treated as a single IP network or subnetwork. Map entries for protocol translators with addresses on subnetworks other than the one on which the interface's IP address is stored are ignored by the routing software. Additionally, all protocol translators using the subnet number should have map entries for all others. There are also issues with the broadcast flag, which apply both to IP and to other protocols with dynamic routing.

```
interface serial 2
ip address 131.108.9.1 255.255.255.0
!
encapsulation X25
!
! The ”bandwidth” command is not part of the X.25
! configuration; it's especially important to understand that it doesn't
! have any connection with the X.25 entity of the same name.
!”bandwidth” commands are used by IP routing processes (currently only IGRP),
! to determine which lines are the best choices for traffic.
```

```
! Since the default is 1544, and X.25 service at that rate isn't generally
! available, most X.25 interfaces that are being used with IGRP in a
! real environment will have ”bandwidth” settings.
!
! This is a 9.6 Kbaud line:
!
bandwidth 10
!
! These Level 3 parameters are defaults; they need to
! match the PDN defaults. They are negotiable on a per-call basis:
!
x25 win 7
x25 wout 7
x25 ips 512
x25 ops 512
!
! You must specify an X.121 address to be assigned to the X.25
! interface by the PDN.
!
x25 address 31370054065
!
! The following Level 3 parameters have been set to match the network.
! You generally need to change some Level 3 parameters, most often
! those listed below. You may not need to change any Level 2
! parameters, however.
!
x25 htc 32
x25 idle 5
x25 nvc 2
!
! The following commands configure the X.25 map. If you want to exchange
! routing updates with any of the protocol translators, they would need
! ”broadcast” flags.
! If the X.25 network is the only path to them, static routes are
! generally used to save on packet charges. If there is a redundant
! path, it might be desirable to run a dynamic routing protocol.
!
x25 map IP 131.108.9.3 31370019134 ACCEPT-REVERSE
! (ACCEPT-REVERSE allows collect calls)
x25 map IP 131.108.9.1 31370054065
x25 map IP 131.108.9.2 31370053087
!
! If the PDN cannot handle fast back-to-back packets, use the
!”transmitter-delay” command to slow down the interface:
!
transmitter-delay 1000
!
```

Example of Virtual Circuit Ranges

The following example sets the following VC ranges: 5 to 20 dedicated to incoming calls only (from the DCE to the DTE), 25 to 1024 for either incoming or outgoing calls, no VC range dedicated to outgoing calls (from the DTE to the DCE). Up to four permanent virtual circuits can be defined on VCs 1 through 4.

```
x25 lic 5
x25 hic 20
x25 ltc 25
```

Example of Switching a PVC on the Same Protocol Translator

In the following example, there is a PVC connected between two serial interfaces on the same protocol translator. In this type of interconnection configuration, the alternate interface must be specified along with the PVC number on that interface. To make a working PVC connection, two commands must be specified, each pointing to the other.

```
interface serial 0
encapsulation x25
x25 ltc 5
x25 pvc 1 interface serial 1 pvc 4
interface serial 1
encapsulation x25
x25 ltc 5
x25 pvc 4 interface serial 0 pvc 1
```

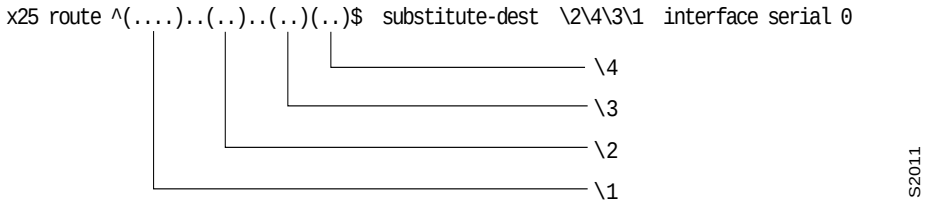
Example of X.25 Route Address Pattern Matching

The following example shows how to indicate that X.25 calls to addresses whose first four Data Network Identification Code (DNIC) digits are 1111 should be routed through interface serial 3, but that the DNIC field in the addresses presented to the equipment connected to that interface should be changed to 2222. The \1 characters in the rewrite pattern indicate the portion of the original address matched by the characters. The asterisk should be inserted in the rewritten address.

```
x25 route ^1111(.*) substitute-dest 2222\1 interface serial 3
```

Figure 1-3 shows a more explicit command intended to illustrate the power of the rewriting scheme.

Figure 1-3 Example of Rewrite Pattern



This scheme causes all X.25 calls with 14-digit called addresses to be routed through interface serial 0. The incoming DNIC field would be moved to the end of the address. The fifth, sixth, ninth, and tenth digits would be deleted, and the thirteenth and fourteenth would be moved before the eleventh and twelfth.

Example of X.25 Routing

The following examples illustrate how to enable an X.25 switch, how to enable call forwarding, and how to configure a protocol translator on a Tymnet/PAD switch to accept and forward calls.

This first example shows how to enable X.25 switching, as well as how to enter routes into the X.25 routing table:

```
!
! Enable X.25 forwarding
x25 routing
!
! Enter routes into the table. Without a positional parameter, entries
! are appended to the end of the table
x25 route ^100$ interface serial 0
x25 route 100 cud ^pad$ interface serial 2
x25 route 100 interface serial 1
x25 route ^3306 interface serial 3
x25 route .* ip 10.2.0.2
!
```

The routing table forwards calls for X.121 address 100 out interface serial 0. Otherwise, if the X.121 address contains 100 anywhere within it and contains no Call User Data, or the Call User Data is not the string pad, it is forwarded onto serial 1. If the X.121 address contains the digits 100 and the Call User Data is the string pad, the call is forwarded onto serial 2. All X.121 addresses that do not match the first three routes are checked for a DNIC of 3306 as the first four digits. If they do match, they are forwarded over serial 3. All other X.121 addresses will match the fifth entry, which is a match-all pattern and will have a TCP connection established to the IP address 10.2.0.2. The protocol translator at 10.2.0.2 will then route the call according to its X.25 routing table.

The following example configures a protocol translator that sits on a Tymnet PAD/switch to accept calls and have them forwarded to a Digital VAX system. This feature permits running X.25 network over a generalized, existing IP network, thereby making it unnecessary to get another physical line for one protocol. The protocol translator positioned next to the Digital VAX system is configured with X.25 routes, as follows:

```
x25 route vax-X.121-address interface serial 0
x25 route .* ip cisco-on-tymnet-ipaddress
```

This configuration routes all calls to the VAX X.121 address out to serial 0, where the VAX is connected running PSI. All other X.121 addresses are forwarded to the *cisco-on-tymnet* address using its IP address. This takes all outgoing calls from the VAX and sends them to *cisco-on-tymnet* for further processing.

On the protocol translator named *cisco-on-tymnet*, you would enter these commands:

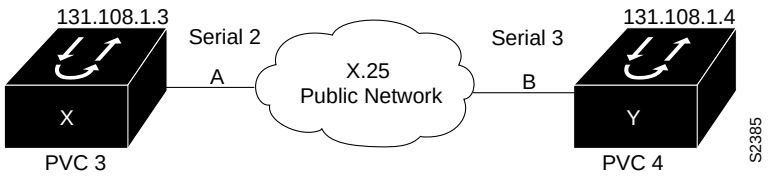
```
x25 route vax-X.121-address ip cisco-on-vax
x25 route .* interface serial 0
```

These commands force all calls with the VAX X.121 address to be sent to the protocol translator with the VAX connected to it. All other calls with X.121 addresses are forwarded out to Tymnet. If Tymnet can route them, a CALL ACCEPTED packet is returned, and everything proceeds normally. If Tymnet cannot handle it, it clears the call, and the CLEAR REQUEST packet is forwarded back toward the VAX.

Example of Using a PVC to Exchange IP Traffic

The following example, illustrated in Figure 1-4, demonstrates how to use the PVC to exchange IP traffic between protocol translator X and protocol translator Y.

Figure 1-4 Establishing an IP Encapsulation PVC through an X.25 Network



Protocol Translator X

```
interface serial 2
ip address 131.108.1.3 255.255.255.0
x25 map ip 131.108.1.4 0
x25 pvc 3 ip 131.108.1.4
```

Protocol Translator Y

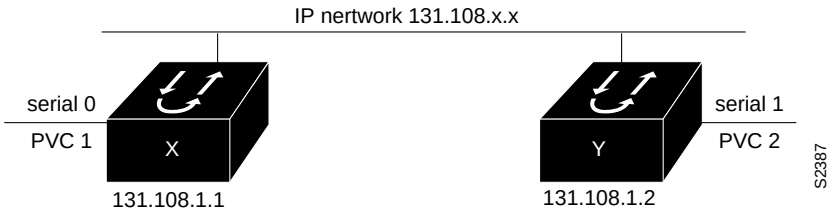
```
interface serial 3
ip address 131.108.1.433 255.255.255.0
x25 map ip 131.108.1.3 0
x25 pvc 4 ip 131.108.1.3
```

In this example, the PDN has established a PVC through its network connecting PVC number 3 of access point A to PVC number 4 of access point B. On protocol translator X, a connection is established between protocol translator X and protocol translator Y's IP address, 131.108.1.4. On protocol translator Y, a connection is established between protocol translator Y and protocol translator X's IP address, 131.108.1.3.

Example of Establishing a Connection between Two PVCs

In the following example, a connection is established between two PVCs across a LAN. Because the connection is remote (across the LAN), the tunneling command is used. This example establishes a PVC between protocol translator X, Serial 0, PVC1 and protocol translator Y, Serial 1, PVC 2. Figure 1-5 provides a visual representation of the configuration.

Figure 1-5 X.25 Tunneling Connection



Protocol Translator X

```
interface serial 0
x25 pvc 1 tunnel 131.108.1.2 interface serial 1 pvc 2
```

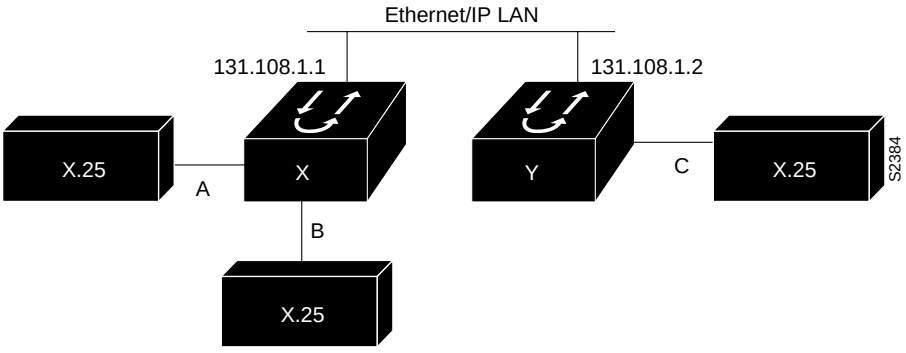
Protocol Translator Y

```
interface serial 1
x25 pvc 2 tunnel 131.108.1.1 interface serial 0 pvc 1
```

Example of Remote Tunneling

In Figure 1-6, the connection between points A and B is switched, and the connections between points A or B and C are tunneled.

Figure 1-6 Local Switching and Remote Tunneling PVCs



Protocol Translator X

```
interface ethernet 0
ip address 131.108.1.1 255.255.255.0
!
interface serial 0
x25 ltc 5
x25 pvc 1 interface serial 1 pvc 1
x25 pvc 2 tunnel 131.108.1.2 interface serial 0 pvc 1
!
interface serial 1
x25 ltc 5
x25 pvc 1 interface serial 0 pvc 1
x25 pvc 2 tunnel 131.108.1.2 interface serial 0 pvc 2
```

Protocol Translator Y

```
interface ethernet 0
ip address 131.108.1.2 255.255.255.0
!
interface serial 0
x25 ltc 5
x25 pvc 1 tunnel 131.108.1.1 interface serial 0 pvc 2
x25 pvc 2 tunnel 131.108.1.1 interface serial 1 pvc 2
```

Example of Enabling CMNS for X.121 and MAC Addresses

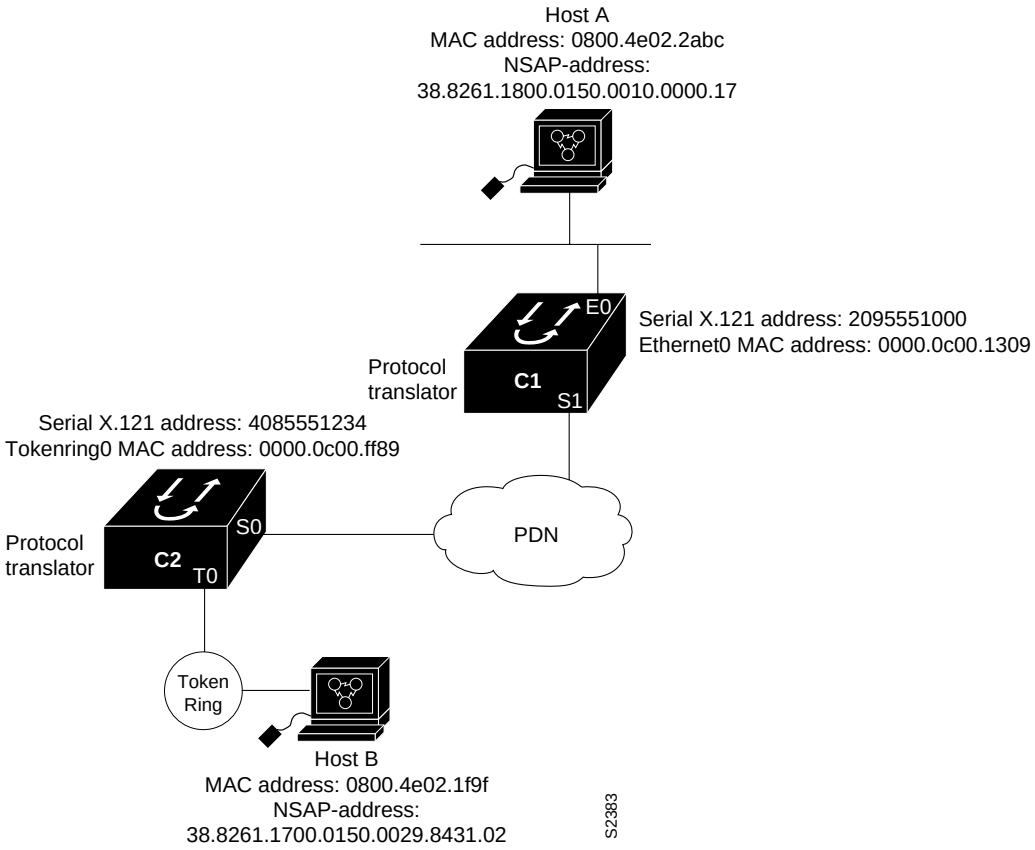
The following example illustrates enabling CMNS and configuring X.121 and MAC-address mappings:

```
interface ethernet 0
cmns enable
x25 map cmns 38.8261.1000.0150.1000.17 0000.0c00.ff89
! Above maps NSAP to MAC-address on Ethernet0
!
interface serial 0
encapsulation x25
x25 map cmns 38.8261.1000.0150.1000.18 3110451
! Above maps NSAP to X.121-address on Serial0
! assuming the link is over a PDN
!
interface serial 1
encapsulation x25
x25 map cmns 38.8261.1000.0150.1000.20
! Above specifies cmns support for Serial1
! assuming that the link is over a leased line
```

Example of Switching CMNS over a PDN

The following example depicts switching CMNS over a packet-switched public data network (PDN). Figure 1-7 illustrates the general network topology for a CMNS switching application where calls are being made between resources on opposite sides of a remote link to Host A (on an Ethernet) and Host B (on a Token Ring), with a PDN providing the connection.

Figure 1-7 Example Network Topology for Switching CMNS over a PDN



The following sample configuration allows resources on either side of the PDN to call Host A or Host B. This configuration allows traffic intended for the remote NSAP address specified in the **x25 map cmns** commands (for the serial ports) to be switched through the serial interface for which CMNS is configured.

The CMNS-related configuration for protocol translator C2 in Figure 1-7 would be as follows:

```
! This configuration specifies that any traffic from any other
! interface intended for any NSAP address with NSAP prefix 38.8261.17
! will be switched to MAC address 0800.4e02.1f9f
! through Token Ring 0
!
interface token 0
cmns enable
x25 map cmns 38.8261.17 0800.4e02.1f9f
!
! This configuration specifies that traffic from any other interface
! on Cisco Protocol translator C2 that is intended for any NSAP address with
! NSAP-prefix 38.8261.18 will be switched to
! X.121 address 2095551000 through Serial 0
!
interface serial 0
encapsulation x25
x25 address 4085551234
x25 map cmns 38.8261.18 2095551000
```

The CMNS-related configuration for protocol translator C1 in Figure 1-7 would be as follows:

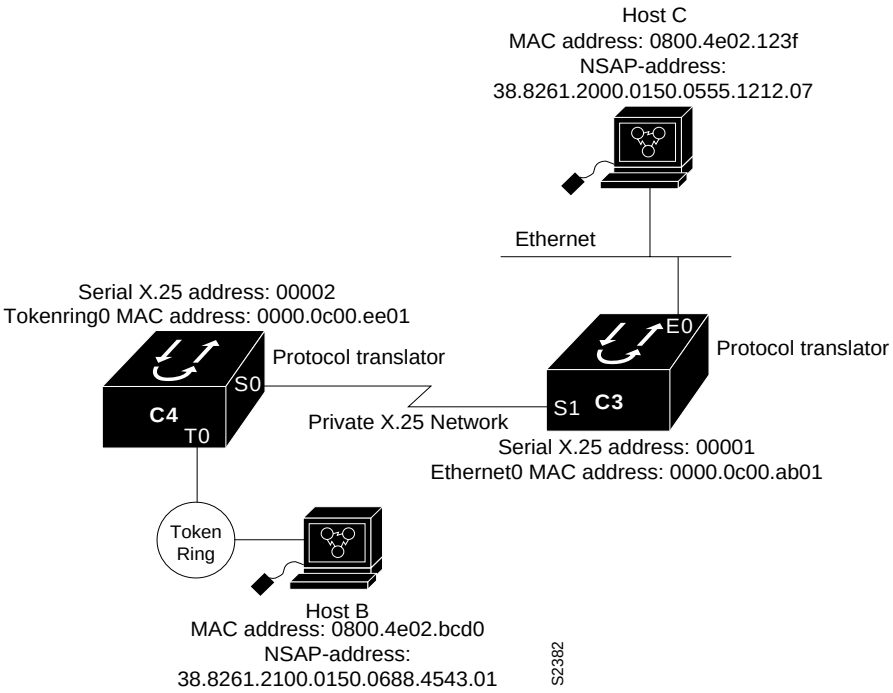
```
! This configuration specifies that any traffic from any other
! interface intended for any NSAP address with NSAP 38.8261.18
! will be switched to MAC address 0800.4e02.2abc through Ethernet 0
!
interface ethernet 0
cmns enable
x25 map cmns 38.8261.18 0800.4e02.2abc
!
! This configuration specifies that traffic from any other interface
! on Cisco Protocol Translator C1 that is intended for any NSAP address with
! NSAP-prefix 38.8261.17 will be switched to X.121 address
! 4085551234 through Serial 1
!
interface serial 1
encapsulation x25
x25 address 2095551000
x25 map cmns 38.8261.17 4085551234
```

Example of Switching CMNS over Leased Lines

The following example illustrates switching CMNS over a leased line. Figure 1-8 illustrates the general network topology for a CMNS switching application where calls are being made by resources on the opposite sides of a remote link to Host C (on an Ethernet) and Host D (on a Token Ring), with a dedicated leased line providing the connection.

The following configuration listing allows resources on either side of the leased line to call Host C or Host D. This configuration allows traffic intended for the remote NSAP address specified in the **x25 map cmns** commands (for the serial ports) to be switched through the serial interface for which CMNS is configured.

Figure 1-8 Example Network Topology for Switching CMNS over a Leased Line



A key difference for this configuration compared with the previous example is that with no PDN, the specification of an X.121 address in the **x25 map cmns** command is not necessary. The specification of an X.25 address also is not needed, but is included for symmetry with the previous example.

The CMNS-related configuration for protocol translator C4 in Figure 1-8 would be as follows:

```
! This configuration specifies that any traffic from any other
! interface intended for any NSAP address with NSAP 38.8261.21
! will be switched to MAC address 0800.4e02.bcd0 through Token Ring 0
!
interface token 0
cmns enable
x25 map cmns 38.8261.21 0800.4e02.bcd0
!
! This configuration specifies that traffic from any other interface
! on Cisco Protocol translator C4 that is intended for any NSAP address with
! NSAP-prefix 38.8261.20 will be switched through Serial 0
!
interface serial 0
encapsulation x25
x25 address 00002
x25 map cmns 38.8261.20
```

The CMNS-related configuration for protocol translator C3 in Figure 1-8 would be as follows:

```
! This configuration specifies that any traffic from any other
! interface intended for any NSAP address with NSAP 38.8261.20
¡ will be switched to MAC address 0800.4e02.123f through Ethernet 0
!
interface ethernet 0
cmns enable
x25 map cmns 38.8261.20 0800.4e02.123f
!
! This configuration specifies that traffic from any other interface
! on Cisco Protocol translator C3 that is intended for any NSAP address with
! NSAP-prefix 38.8261.21 will be switched through Serial 1
!
interface serial 1
encapsulation x25
x25 address 00001
x25 map cmns 38.8261.21
```

Example of DDN X.25 Configuration

The following example illustrates how to configure a protocol translator interface to run DDN X.25:

```
interface serial 0
ip address 192.31.7.50 255.255.255.240
encapsulation DDNX25
x25 win 6
x25 wout 6
x25 ips 1024
x25 ops 1024
x25 t20 10
x25 t21 10
x25 t22 10
x25 t23 10
x25 nvc 2
x25 map IP 192.31.7.49 000000010300 BROADCAST
```

Example of BFE Emergency Mode

In the following example, interface Serial 0 is configured to require an EXEC command from the administrator before it participates in emergency mode. The host IP address is 21.0.0.12, and the address of the remote BFE unit is 21.0.0.1. When the BFE enters emergency mode, the protocol translator will prompt the administrator for EXEC command **bfe enter** to direct the protocol translator to participate in emergency mode.

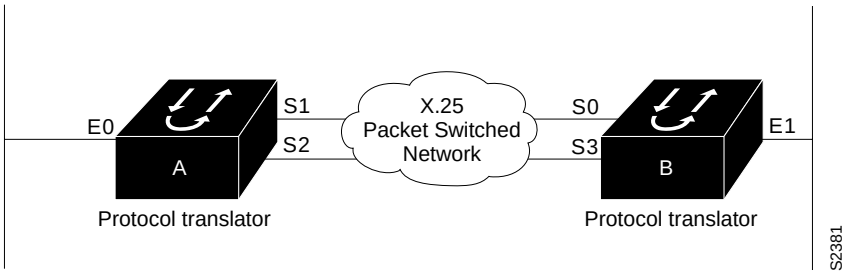
```
interface serial 0
ip address 21.0.0.2 255.0.0.0
encapsulation bfex25
x25 bfe-emergency decision
x25 remote-red 21.0.0.12 remote-black 21.0.0.1
x25 bfe-decision ask
```

Example of Configuring X.25 to Allow Ping Support over Multiple Lines

For **ping** commands to work in an X.25 environment (when load sharing over multiple serial lines), you must include entries for all adjacent interface IP addresses in the **x25 map** command for each serial interface. The following example illustrates this point.

Consider two protocol translators, A and B, communicating with each other over two serial lines via an X.25 PDN (see Figure 1-9) or over leased lines. In either case, all serial lines must be configured for the same IP subnet address space. In order to allow for successful **ping** commands, the configuration might be as in the lists that follow. In any event, a similar configuration is required for the same subnet IP addresses to work across X.25.

Figure 1-9 Parallel Serial Lines to X.25 Network



Note All four serial ports configured for the two protocol translators in the following configuration example must be assigned to the same IP subnet address space. In this case, the subnet is 131.108.170.0.

Configuration for Protocol Translator A

```
!
interface serial 1
ip 131.108.170.1 255.255.255.0
x25 address 31370054068
x25 map ip 131.108.170.3 31370054065
x25 map ip 131.108.170.4 31370054065

interface serial 2
ip 131.108.170.2 255.255.255.0
x25 address 31370054069
x25 map ip 131.108.170.4 31370054067
x25 map ip 131.108.170.3 31370054067
! allow either source address
x25 31370054068 alias serial2
x25 31370054069 alias serial1
```

Configuration for Protocol Translator B

```
!
interface serial 0
ip 131.108.170.3 255.255.255.0
x25 address 31370054065
x25 map ip 131.108.170.1 31370054068
x25 map ip 131.108.170.2 31370054068

interface serial 3
ip 131.108.170.4 255.255.255.0
x25 address 31370054067
x25 map ip 131.108.170.2 31370054069
x25 map ip 131.108.170.1 31370054069
! allow either source address
x25 31370054065 alias serial3
x25 31370054067 alias serial0
```

Example of Netbooting over X.25

When netbooting over X.25, you cannot netboot via a broadcast. You must netboot from a specific host. Also, an **x25 map** command must exist for the host that you netboot from. In the following example, if file gs3-bfx is to be booted from a host with IP address 131.108.126.2, the following information would need to be in the configuration:

```
boot system gs3-bfx 131.108.126.2
interface Serial 0
encapsulation x25
x25 map IP 131.108.126.2 10001 broadcast
```

The **x25 map** command is used to map an IP address into an X.121 address. There must be an **x25 map** command that matches the IP address given on the **boot system** command line. The following is an example of such a configuration:

```
boot system gs3-bfx.83-2.0 131.108.13.111
!
interface Serial 1
ip address 131.108.126.200 255.255.255.0
encapsulation X25-DCE
x25 address 10004
x25 map IP 131.108.13.111 10002 broadcast
lapb n1 12040
clockrate 56000
```

In this case, 10002 is the X.121 address of the remote protocol translator that can get to host 131.108.13.111.

The remote protocol translator must have the following **x25 map** entry:

```
x25 map IP 131.108.126.200 10004 broadcast
```

This allows the remote protocol translator to return a boot image (from the netboot host) to the protocol translator netbooting over X.25.

Example of an X.29 Access List

The following example illustrates an X.29 access list. Incoming permit conditions are set for all IP hosts and LAT nodes that have specific characters in their names. All X.25 connections to a printer are denied. Outgoing connections are restricted.

```
!
!Permit all IP hosts and LAT nodes beginning with “VMS”.
!Deny X.25 connections to the printer on line 5.
!
access-list 1 permit 0.0.0.0 255.255.255.255
lat access-list 1 permit ^VMS.*
x29 access-list 1 deny .*
!
line vty 5
access-class 1 in
!
!Permit outgoing connections for other lines.
!
!Permit IP access with the network 131.108
access-list 2 permit 131.108.0.0 0.0.255.255
!
!Permit LAT access to the boojum/snark complexes.
lat access-list 2 permit ^boojum$
lat access-list 2 permit ^snark$
!
!Permit X.25 connections to Infonet hosts only.
x29 access-list 2 permit ^31370
!
line vty 0 16
access-class 2 out
!
translate tcp 131.108.1.26 x25 5551234 access-class 2
```

Example of an X.3 Profile

The following profile script turns local edit mode on when the connection is made and establishes local echo and line termination upon receipt of a Return. The name “linemode” is used with the **translate** command to effect use of this script.

```
x29 profile linemode 2:1 3:2 15:1
translate tcp 131.108.1.26 x25 55551234 profile linemode
```

The X.3 PAD parameters are described in Appendix F of the *Protocol Translator Command Reference* and in the *Communication Server and Protocol Translator Connection Guide*.